Abstract

New PCI-e flash cards and SSDs supporting over 100,000 IOPs are now available, with several usecases in the design of a high performance storage system. By using an array of flash chips, arranged in multiple banks, large capacities are achieved. Such multi-banked architecture allow parallel read, write and erase operations. In a raw PCI-e flash card, such parallelism is directly available to the software layer. In addition, the devices have restrictions such as, pages within a block can only be written sequentially. The devices also have larger minimum write sizes (>4KB). Current flash translation layers (FTLs) in Linux are not well suited for such devices due to the high device speeds, architectural restrictions as well as other factors such as high lock contention. We present a FTL for Linux that takes into account the hardware restrictions, that also exploits the parallelism to achieve high speeds. We also consider leveraging the parallelism for garbage collection by scheduling the garbage collection activities on idle banks. We propose and evaluate an adaptive method to vary the amount of garbage collection according to the current I/O load on the device.

For large scale distributed storage systems, flash memories are an excellent choice because flash memories consume less power, take lesser floor space for a target throughput and provide faster access to data. In a traditional distributed filesystem, even distribution is required to ensure load-balancing, balanced space utilisation and failure tolerance. In the presence of flash memories, in addition, we should also ensure that the number of writes to these different flash storage nodes are evenly distributed, to ensure even wear of flash storage nodes, so that unpredictable failures of storage nodes are avoided. This requires that we distribute updates and do garbage collection, across the flash storage

nodes. We have motivated the distributed wearlevelling problem considering the replica placement algorithm for HDFS. Viewing the wearlevelling across flash storage nodes as a distributed co-ordination problem, we present an alternate design, to reduce the message communication cost across participating nodes. We demonstrate the effectiveness of our design through simulation.